Theoretical Computer Science 46 (1986) 197–217
North-Holland

COMPUTABILITY IN HIGHER TYPES, $P\omega$ AND THE COMPLETENESS OF TYPE ASSIGNMENT*

G. LONGO and S. MARTINI

Dipartimento di Informatica, Università di Pisa, I-56100 Pisa, Italy

Communicated by E. Engeler
Received June 1983
Revised February 1986

Abstract. $P\omega$, the powerset of the natural numbers, may be turned into an applicative structure by Myhill and Shepherdson, "$\cdot$". Then, for $A, B \subseteq P\omega$, set $A \to B = \{d \in P\omega \mid \forall a \in A, da \in B\}$. Any effectively given domain (in the sense of Scott (1982)) can be embedded into $P\omega$ by a continuous and computable retraction (*notation*: $X \lhd_c A_X$, for some $A_X \subseteq P\omega$, which is also an effectively given domain). We first prove that if $X \lhd_c A_X$ and $Y \lhd_c A_Y$, then also $A_X \to A_Y$ is an effectively given domain and ($\star$): $\mathrm{Cont}(X, Y) \lhd_c A_X \to A_Y$, i.e., the continuous functions can be embedded into $A_X \to A_Y$.

Let now $P \subseteq P\omega$ be the collection of single-valued sets, i.e., P is isomorphic to the effectively given domain of the partial functions on ω, and let T be the function-type symbols, with $(1) \in T$. Then, for $P^{(1)} = P$, $P^{\sigma\to\tau} = P^\sigma \to P^\tau$ extends the classical recursive operators at higher types. By ($\star$), Ershov's model of the Kleene–Kreisel countable functionals can effectively be embedded, by some G_σ's, into the type structure $\{P^\sigma\}_{\sigma\in T}$ in $P\omega$. Thus, the recursive functionals correspond to the r.e. sets in the due types, for example, f has type $\sigma \to \tau$ iff $G_{\sigma\to\tau}(f)$ is an r.e. set in $P^\sigma \to P^\tau$.

$\{P^\sigma\}_{\sigma\in T}$ clearly yields a model for formal type assignment to terms of λ-calculus, i.e., for any assignment B of types to variables and any $\sigma \in T$ one has $B \vdash \sigma M \Rightarrow [\![M]\!]_{\xi_B} \in P^\sigma$, where $\xi_B : \mathrm{Var} \to P\omega$, according to B.

We prove that also the reverse implication holds for typable terms. Thus, a completeness theorem for type checking is established over a model defined by an independent recursion-theoretical motivation.

Introduction

In 1955, a pioneering paper by Myhill and Shepherdson [36] gave a good momentum to the study of Recursion Theory in higher types. They defined a natural notion of effective operation over number-theoretic functions and characterized it by using a very fruitful notion of application "$\cdot$" in $P\omega$ (see below). Shortly thereafter, papers by Gödel [18], Kleene [26] and Kreisel [27] extended Recursion Theory to any finite type over ω (i.e. functions over ω, functionals on these functions and so on: the "Type Structure" over ω) (see [37]). Kleene, Gödel and Kreisel's work was also motivated by consistency results for intuitionistic arithmetic and constructive mathematics.

* A preliminary version of this paper (Sections 1 and 2) appeared in the Springer Lecture Notes in Computer Science, Vol. 166. This research was partially supported by the Ministry of P.I. (40%).

For the purposes of Computer Science, (effective) transformations over number-theoretic functions are surely interesting; they are (or interpret) programs acting over programs. Nevertheless, from the computer scientist's point of view, one should be able to (effectively) deal with different and various sorts of data also. The point then is to arrange the data in structures where interesting notions of (effective) operations may be given. This motivated the work of Scott, some 10–15 years ago, on posets (lattices, in particular). Since then, computability in abstract structures turned out to be a fruitful research area (see, for example, [14, 25, 44, 45] for work in various directions). Scott [43] unifies terminology and approaches.

Moreover, the 'abstract' approach also provided an insight into Recursion Theory in higher types. As a matter of fact, Ershov (see [15, 16], and a lot more in Algebra and Logic from 1972 to 1976 and in ZML, 1975) and Hyland [24], by using cartesian closed categories (ccc's) of (effective) spaces, elegantly characterized the Kleene–Kreisel countable functionals, as sub-ccc generated by the object ω (roughly speaking, see [30] for a discussion).

Scott's domains and Ershov's complete f_0-spaces are readily seen to be equivalent (see [17] for a discussion and [31] for some recent recursion-theoretic applications). Besides the motivations and the aims, Scott, Ershov and Hyland's approaches also use a common technique, which may be shortly described and related to other techniques in mathematics as follows. Taken for granted the notion of recursive function (r.e. set), by 'local', topological or convergence properties the notion of computability is extended to abstract spaces: an element is computable when its neighbourhoods' filter has an r.e. representation (see [30] for a discussion). By a very informal analogy, something similar is done for 'differentiable manifolds'. Given the good old notion of differentiability over R^n, by a system of local coordinates one defines differentiability in abstract spaces.

For increasing types, though, the intuition of the structure of the 'neighbourhoods' filter' gets more and more vague. The purpose of this paper is to 'take back' to Myhill–Shepherdson's $\langle P\omega, \cdot \rangle$ as much as possible of the abstract approach. An (effective) functional of a given type over an arbitrary domain will be represented by the application "$\cdot$" and an (r.e.) set in a corresponding 'type' as a subset of $P\omega$ (Lemma 2.2). This approach was first proposed in [28] and was inspired by Scott's semantics of Curry's type assignment system and the work done on Extended Type Structures in [10].

The type structure corresponding to the higher type functionals in $P\omega$ will be denoted by $\{P^\sigma\}_{\sigma \in T}$, where $P^{(1)} \subseteq P\omega$ are the (graphs of) the partial functions on ω. The last section gives a further motivation for looking at $\{P^\sigma\}_{\sigma \in T}$.

As is well known, $(P\omega, \cdot)$ yields a model for a paradigmatic type-free functional language: the λ-calculus. The type discipline for λ-calculus has been widely explored in Computer Science (e.g., [35, 33, 9]). Henderson [19] relates type-checking of most programming languages constructs to type-checking of a functional language such as λ-calculus. The point with formal type-checking is its semantic soundness, in order to prove the consistency of the type discipline, and, possibly, its completeness

w.r.t. the given semantics. By using the classical type assignment system of Curry for typed λ-calculus, we prove that $\{P^\sigma\}_{\sigma\in T}$ give a sound and complete model for typable λ-terms (w.r.t. the simple semantics; see [21]).

Moreover, we show the completeness w.r.t. the F-semantics (Remark 4.9). Thus, our 'recursion-theoretic' hierarchy also yields a model for the completeness of type-checking, where each function has a unique representative.

1. Domains and $P\omega$

For the notion of domain we refer to Scott [43]. In a poset $(X, \leqslant)$ set $\check{x} = \{y\in X \mid x\leqslant y\}$ for short; then a *domain* is an algebraic c.p.o. $(X, X_0, \leqslant)$, where $X_0 = \{x\in X \mid \check{x} \text{ is open in Scott topology}\}$, the set of compact elements, and X_0 has *bounded joins*. That is, if $x_0, y_0 \in X_0$ are compatible (i.e., $\exists z\in X, x_0, y_0 \leqslant z$; *notation*: $x_0 \uparrow y_0$), then $x_0 \sqcup y_0$ exists and is in X_0. An *effectively given domain* $(X, X_0, \nu, \leqslant)$ is a countably based domain such that, for the given numbering $\nu: \omega \to X_0$, $\nu(n)\uparrow\nu(m)$ is decidable in n, m and $\nu(n) = \nu(p)\sqcup\nu(q)$ is decidable in n, p, q. An element x of an effectively given domain $(X, X_0, \nu, \leqslant)$ is *computable* if $\{n \mid \nu(n)\leqslant x\}$ is r.e. (*notation*: $x\in X_c$). As a matter of fact, Scott [43] introduces domains as the completion over filters of neighbourhood systems. The equivalence with the above definition can readily be seen (cf. [17]).

Of course, for a given canonical numbering $e: \omega \to P\omega_0$ of the finite sets, $(P\omega, P\omega_0, e, \subseteq)$ is an effectively given domain.

The category of (effectively given) domains is cartesian closed with continuous functions (w.r.t. Scott topology) as morphisms (*notation*: $\mathrm{Cont}(X, Y)$ are the continuous functions from X to Y). Thus, the notions of compact and computable elements are inherited at higher types. In particular, $\mathrm{Cont}(X, Y)_c$ are the continuous and computable maps from X to Y.

Notation. (i) $\langle\ ,\ \rangle: \omega^2 \leftrightarrow \omega$ is a bijective pairing and $\{e_n\}_{n\in\omega} = P\omega_0$ is an effective numbering of the finite subsets of ω.

(ii) Given domains $(X, X_0, \leqslant)$ and $(Y, Y_0, \leqslant)$ set step $xy = \lambda z.$(**if** $x\leqslant z$ **then** y **else** $\bot$); step xy is continuous if $x\in X_0$. As is well known, the elements of $\mathrm{Cont}(X, Y)_0$ are exactly the finite sups of compatible step functions over X_0, Y_0. If (X, X_0, ν) and (Y, Y_0, μ) are effectively given, then for $g_n \in \mathrm{Cont}(X, Y)_0$ one has $g_n = \bigsqcup_{\langle i,j\rangle\in e_{k(n)}} \text{step } \nu(i)\mu(j) = \bigsqcup_{e_{k(n)}} \text{step } \nu(i)\mu(j)$, where $k: \omega \to \omega$ is a recursive function such that $e_{k(n)} = e_n$ if, for every $J\subseteq e_n$, $J = \{\langle i_1, j_1\rangle, \ldots, \langle i_p, j_p\rangle\}$, one has step $\nu(i_1)\mu(j_1)\uparrow\cdots\uparrow$ step $\nu(i_p)\nu(j_p)$, and $e_{k(n)} = \emptyset$ otherwise.

Lemma 1.1. *Let ν and μ be the given numbering of X_0 and Y_0, respectively. Then $f\in \mathrm{Cont}(X, Y)_c$ iff $f\in \mathrm{Cont}(X, Y)$ and $\mu(j)\leqslant f(\nu(i))$ is semidecidable in i, j.*

Proof. Let $\{g_n\}_{n\in\omega} = \mathrm{Cont}(X, Y)_0$. Then $g_n = \bigsqcup_{e_{k(n)}} \mathrm{step}\ \nu(i)\mu(j) \leq f$ iff $\forall\langle i,j\rangle \in e_{k(n)}$, $\mu(j) \leq f(\nu(i))$. The result now easily follows. □

Any domain $(X, X_0, \leq)$ induces a structure on its subsets by the induced Scott topology. That is, we can make the following remark.

Remark 1.2. Let $(X, X_0, \leq)$ be a domain and $Y \subseteq X$. Define $Y_0 = \{y \in Y \mid \check{y} \subseteq Y$ is open in the induced topology$\}$. Then $(Y, Y_0, \leq)$ is a domain iff

(1) Y is a (sub-)c.p.o.,

(2) Y_0 has bounded joins (w.r.t. Y), and

(3) $(\forall y, y' \in Y)(y \not\leq y' \Rightarrow \exists y_0 \in Y_0 y_0 \leq y$ and $y_0 \not\leq y')$.

Lemma 1.3. *Let $(X, X_0, \leq)$ be a domain. Then any closed subset of X, with the induced structure, is a domain.*

Proof. Let Y be a closed subset of X. Then, $y \in Y$ **and** $y' \leq y \Rightarrow y' \in Y$ and $x \notin Y \Rightarrow \exists x_0 \in X_0 x_0 \notin Y$ **and** $x_0 \leq x$, since $\bar{Y}$ is open and $\{x_0 \mid x_0 \in X_0\}$ is a basis for the topology on X. Thus, conditions (1), (2), (3) of Remark 1.2 easily follow. □

Of course, if $Y \subseteq X$ is closed, $Y_0 = X_0 \cap Y$.

Definition 1.4. Let $(X, X_0, \nu, \leq)$ be an effectively given domain. $Y \subseteq X$ is *effective* if $\nu(n) \in Y$ is decidable in n.

Clearly, any closed effective subset Y of an effectively given domain X is also an effectively given (sub-)domain. Moreover, $Y_c = Y \cap X_c$.

Recall now that, for sets X and Y, a *retraction* (f, g) is a pair $g: X \to Y$ and $f: Y \to X$ such that $f \circ g = \mathrm{id}_X$ (*notation*: $X \lhd Y$, via (g, f)). In case X and Y are effectively given domains and $X \lhd Y$ via some continuous and computable functions f and g, we write $X \lhd_c Y$. We next show that any effectively given domain is a continuous and computable retraction of an effectively given subdomain of $P\omega$ (see also Remarks 1.6).

Theorem 1.5. *Let $(X, X_0, \nu, \leq)$ be an effectively given domain. Then there exists $A_X \subseteq P\omega$ closed and effective such that $X \lhd_c A_X$.*

Proof. For the sake of simplicity, write x_n for $\nu(n)$ in X_0. By assumption, X is a countably based T_0-space. Thus, we can use the embedding $G: X \to P\omega$ given in [42], i.e., $G(x) = \{n \mid x_n \leq x\}$. Define then

$$A_X = \{a \in P\omega \mid \exists x \in X a \subseteq G\{x\}\}$$

and

$$F: A_X \to X \text{ by } F(a) = \bigsqcup \{x_n \mid n \in a\}.$$

Claim 1.5.1. *F is well defined and $F \circ G = \mathrm{id}_X$.* Just notice that $\forall a \in A_X$, $\exists x \in X$, $\forall n \in a$, $x_n \leq x$ and then $F(a) = \bigsqcup \{x_n \mid n \in a\}$ exists in X. $F(G(x)) = x$ is immediate.

Claim 1.5.2. *A_X is closed.* A_X is clearly downward closed. Take now an arbitrary directed set $\{e_i\}_{i \in I}$ in A_X. We only need to show that $\bigcup_I e_i \in A_X$. Notice that $F(e_i) = \bigsqcup \{x_n \mid n \in e_i\} \in X_0$ and that $\forall i, j \in I$, $F(e_i) \sqcup F(e_j) = F(e_i \cup e_j)$. Thus, $x = \bigsqcup \{F(e_i) \mid i \in I\} \in X$ exists and $\bigcup_I e_i \subseteq G(x) \in A_X$. This proves Claim 1.5.2.

Claim 1.5.3. *A_X is effective.* In fact, $e_n \in A_X \Leftrightarrow \exists z, \forall p \in e_n, x_p \leq z$, which is decidable. By the claims and Lemma 1.3, A_X is an effectively given domain.

Claim 1.5.4. *$G \in \mathrm{Cont}(X, A_X)_c$ and $F \in \mathrm{Cont}(A_X, X)_c$, i.e., G and F are continuous and computable.* The proof of the continuity is a simple exercise. Let now $\{e_n^A\}_{n \in \omega}$ be a numbering of $(A_X)_0 = P\omega_0 \cap A_X$. Then, $e_n^A \subseteq G(x_m)$ is clearly decidable in n, m and Lemma 1.1 applies. Similarly for F. □

Remarks 1.6. (i) Let X, A_X, G and F be as in Theorem 1.5 (and its proof). Clearly, $G \circ F \geq \mathrm{id}_{A_X}$. Does $G \circ F$ give a retract $a_X \in P\omega$ (a closure, actually) such that $A_X = \text{range } a_X$ (see [42])? This is not true in general, since the range of any $a_X \in P\omega$ is a lattice, whereas X is a lattice iff A_X is a lattice iff $A_X = P\omega$. Moreover, it is easy to give a domain X such that the corresponding A_X does not admit any $\bar{F} \in \mathrm{Cont}(P\omega, X)$ extending $F \in \mathrm{Cont}(A_X, X)$. (Recall that continuous lattices are exactly the injective spaces (see [41]).)

Therefore, one may define, using Remark 2.1 below, $a_X = \mathrm{Graph}(G \circ F) = \{\langle n, m \rangle \mid m \in G \circ F(e_n)\} \in P\omega_c$ (the r.e. sets) iff X is a lattice.

(ii) The first aim of this paper is to interpret within $P\omega$ the higher type objects over abstract structures, i.e., to interpret in $P\omega$ the space of morphisms (the 'arrow' objects) of some interesting categories. Taking domains which are lattices, by (i), we could use the full strength of the theory of retracts in [42, Section 4] with the corresponding notion of "$\to$" in $P\omega$. We prefer to deal with the more general cases for two reasons. First, in [39], general sound motivations are given for the computational and semantic interest of posets which are not lattices (see also [43]). The domain P, say, of partial functions from ω to ω is not a lattice; the interest of this space is obvious and our main applications will deal with it. Second, by the interpretation of "$\to$" in $P\omega$ used below, our work directly relates to classical recursion theory.

(iii) Another possibility would be to consider Plotkin's universal domain $T\omega$, which is not a lattice. Then, any (effectively given) coherent domain would be (isomorphic to) a retract of $T\omega$, namely the range of an element a_X of $T\omega$, representing, via "$\cdot$" in $T\omega$, the continuous (and computable) function which corresponds to $G \circ F$ above. a_X would also be a closure (and a computable element of $T\omega$, i.e., a pair of disjoint r.e. sets) (see [39, Theorems 11, 20, 21]). Plotkin gives a theory of retracts corresponding to Scott's work for $P\omega$. We would miss though the natural

extension to higher types of Myhill and Shepherdson's recursive operators, which motivated the research in [28, Section 2] and in this paper, as well as the simple notion of 'application' they are based on (see below). As a matter of fact, what is gained by dealing with a c.p.o. instead of a lattice is lost in simplicity and transparency, since the notion of application over $T\omega$ is not so immediate (see [5, 6] for some work on $T\omega$ as a model of type-free λ-calculus and as a tool for type-two recursion theory).

2. Type structure in $P\omega$

Let $a, b \in P\omega$. Define

$$a \cdot b = \{m \mid \exists e_n \subseteq b, \langle n, m\rangle \in a\}.$$

Given $A, B \subseteq P\omega$, now set

$$A \to B = \{d \in P\omega \mid \forall a \in A, da \in B\}.$$

Remark 2.1. (i) For $a \in P\omega$ and $f \in \mathrm{Cont}(P\omega, P\omega)$, set

$$\mathrm{Funct}(a) = \lambda x.ax \in \mathrm{Cont}(P\omega, P\omega) \quad \text{and} \quad \mathrm{Graph}(f) = \{\langle n, m\rangle \mid m \in f(e_n)\}.$$

Then, $\mathrm{Cont}(P\omega, P\omega) \lhd P\omega$, via (Graph, Funct). Both Funct and Graph are continuous. Moreover, given "$\cdot$", Graph is the unique continuous function such that $\mathrm{Graph}(f) \cdot a = f(a)$. In other words, the interpretation of λ-abstraction is unique in $P\omega$ (in [29] this is shown in a general setting).

(ii) Graph and Funct are also computable, in the due types. Let $\mathrm{Cont}(P\omega, P\omega)_0 = \{g_n\}_{n\in\omega}$. Then,

$$g_n = \bigsqcup_{e_{k(n)}} \mathrm{step}\ e_i e_j \leqslant \mathrm{Funct}(e_p) = \bigsqcup_{\langle r,s\rangle \in e_p} \mathrm{step}\ e_r\{s\}$$

is decidable in n, p. Moreover,

$$e_p \subseteq \mathrm{Graph}(g_n) = \left\{\langle m, q\rangle \;\middle|\; q \in \left(\bigsqcup_{e_{k(n)}} \mathrm{step}\ e_i e_j\right)(e_m)\right\}$$

is also decidable in p and n. Thus, Lemma 1.1 applies.

Lemma 2.2. *Let A, B closed (and effective) subsets of $P\omega$. Then, $A \to B$ and $A \cap B$ are closed (and effective).*

Proof. Let $d' \subseteq d \in A \to B$. Then $\forall a \in A$, $d'a \subseteq da \in B$ and, by the assumption, $A \to B$ is downward closed.

Let $d \notin A \to B$. Then, for some $a \in A$, $da \notin B$. By the assumption, $\exists e_n \subseteq da$, $\check{e}_n \subseteq \bar{B}$. Then set

$$e_q = \{\langle p, m\rangle \mid m \in e_n \textbf{ and } p = \min_i [e_i \subseteq a \textbf{ and } \langle i, m\rangle \in d]\}.$$

Clearly, $e_n = e_q a \subseteq da$ and $d \in \check{e}_q \subseteq \overline{A \to B}$. Thus, $A \to B$ is closed.

Finally,

$$e_p \in A \to B \quad \text{iff} \quad \forall J \subseteq e_p \left(\bigcup_{\langle r,s\rangle \in J} e_r \in A \Rightarrow \bigcup_{\langle r,s\rangle \in J} \{s\} \in B \right).$$

(This is trivial in view of $e_p a = \bigcup_{e_n \subseteq a} \{m \mid \langle n, m\rangle \in e_p\}$.) This proves that if A and B are effective, then also $A \to B$ is effective.

The result is immediate for $A \cap B$. □

Definition 2.3. A collection T of *function type symbols* is the least set containing the set AT of atomic types $\phi, \psi, \ldots$ and such that if $\sigma, \tau \in T$, then $\sigma \to \tau \in T$.

The *extended type symbols* are the least set T_E such that $T \subseteq T_E$ and if $\sigma, \tau \in T_E$, then $\sigma \cap \tau \in T_E$.

Remark 2.4. As usual, (higher type) functions of several arguments will be 'embedded' into $P\omega$ using the cartesian closure of the category of (effectively given) domains. That is, by using $\mathrm{Cont}(X \times Y, Z) \cong \mathrm{Cont}(X, \mathrm{Cont}(Y, Z))$ for arbitrary domains X, Y, Z and the relation of $\mathrm{Cont}(\cdot, \cdot)$ to $\cdot \to \cdot$ in $P\omega$ in Lemma 2.10.

Definition 2.5. Let $V: \mathrm{AT} \to PP\omega$ and set $V^\phi = V(\phi)$. Then, extend V to $V: T_E \to PP\omega$ by $V^{\sigma \to \tau} = V^\sigma \to V^\tau$, $V^{\sigma \cap \tau} = V^\sigma \cap V^\tau$.

By induction, Lemma 2.2 immediately gives the following theorem.

Theorem 2.6. *Let $V^\phi \subseteq P\omega$ be closed (and effective) for all $\phi \in \mathrm{AT}$. Then $\forall \sigma \in T_E$, V^σ is closed (and effective).*

The following facts give some structural information on $\{V^\sigma\}_{\sigma \in T_E}$ as a type structure in $P\omega$.

Lemma 2.7

(1) $\forall \phi \in \mathrm{AT}, \emptyset \in V^\phi \Rightarrow \forall \sigma \in T_E, \emptyset \in V^\sigma$.

(2) $\forall \phi \in \mathrm{AT}, \omega \notin V^\phi \Rightarrow \forall \sigma \in T_E, \omega \notin V^\sigma$.

Proof. Notice that $\forall a \in P\omega$, $\emptyset a = \emptyset$ and $\omega a = \omega$. □

Of course, for a closed set V, $V = P\omega$ iff $\omega \in V$.

Proposition 2.8. *Let $V^\phi \neq P\omega$ be closed for all $\phi \in \mathrm{AT}$. Then one has, for all $\sigma, \tau, \rho, \nu \in T_E$:*

$$V^{\sigma \to \tau} \subseteq V^{\rho \to \nu} \Leftrightarrow V^\rho \subseteq V^\sigma \textit{ and } V^\tau \subseteq V^\nu.$$

Proof. ($\Leftarrow$) is an easy exercise and left to the reader.

($\Rightarrow$): We first show $V^\tau \subseteq V^\nu$. Otherwise, let $a \in V^\tau \setminus V^\nu$. Then $\mathrm{Graph}(\lambda x.a) \in V^\sigma \to V^\tau$ and $\mathrm{Graph}(\lambda x.a) \notin V^{\rho\to\nu}$; a contradiction. As for $V^\rho \subseteq V^\sigma$, assume that $b \in V^\rho \setminus V^\sigma$. Since V^ρ and V^σ are closed, for some $e_n \in V^\rho \setminus V^\sigma$, $b \in \check{e}_n \subseteq \overline{V^\sigma}$. Now take $e_m \notin V^\nu$ (e_m exists by Theorem 2.6) and $f = \mathrm{step}\, e_n e_m$. Then, $f \in \mathrm{Cont}(P\omega, P\omega)$ and $\mathrm{Graph}(f) \in V^\sigma \to V^\tau$, while $\mathrm{Graph}(f) \notin V^\rho \to V^\nu$. □

Thus "$\to$" is an injective type constructor, except for $\lambda x.(x \to P\omega)$. (Note that "$\to$" is contravariant in the first argument.) Moreover, if $\forall \phi \in \mathrm{AT}$, $\omega \notin V^\phi$, then $\bigcup_{\sigma \in T_E} V^\sigma \neq P\omega$.

Recall now that any closed (and effective) subset of $P\omega$ is an (effectively given) domain.

Lemma 2.9. *Let* $A \subseteq P\omega$ *and* $f \in \mathrm{Cont}(A, P\omega)$. *Define* $\mathrm{Ext}(f) \in \mathrm{Cont}(P\omega, P\omega)$ *by*

$$\mathrm{Ext}(f)(b) = \bigcup_{e_n \subseteq b} \cap \{f(a) \mid e_n \subseteq a \in A\}.$$

Then, if A is closed (and effective),

$$\mathrm{Ext} \in \mathrm{Cont}(\mathrm{Cont}(A, P\omega), \mathrm{Cont}(P\omega, P\omega))_c.$$

Proof. $\mathrm{Ext}(f)$ is a continuous extension of f. Notice that $\mathrm{Ext}(f)(b) = \bigcup \{f(e_n) \mid e_n \subseteq b$ **and** $e_n \in A\}$, for A is downward closed and f is monotone. Then, the continuity of Ext follows by easy computation. Assume now that A is also effective: let $\{e_n^A\}_{n\in\omega}$ be a numbering of $A_0 = P\omega_0 \cap A$; $\{g_n\}_{n\in\omega} = \mathrm{Cont}(A, P\omega)_0$ and $\{h_n\}_{n\in\omega} = \mathrm{Cont}(P\omega, P\omega)_0$. Recall now that $g_n = \bigsqcup_{e_{k(n)}} \mathrm{step}\, e_i^A e_j^A$, where k is the 'compatibility function' with respect to A. Define then a (total) computable function t such that if $e_{k(n)} = \emptyset$, then $t(n) = k(n)$; and, if $e_{k(n)} \neq \emptyset$, $e_{t(n)}$ is obtained by substituting for every $\langle i, j\rangle \in e_{k(n)}$ the pair $\langle r, s\rangle$ with $e_i^A = e_r$ and $e_j^A = e_s$ (remember that $e_j^A = e_i$ if $e_i \in A$, and $e_i^A = \emptyset$ otherwise). Therefore, $g_n = g_{t(n)}$, $\mathrm{Ext}(g_n) = \mathrm{Ext}(g_{t(n)}) = h_{t(n)}$ and, hence, $h_m \leq h_{t(n)}$ is decidable. Thus, Lemma 1.1 applies and Ext is computable. □

Lemma 2.10. *Let* $A, B \subseteq P\omega$ *be closed (and effective). Then*

$$\mathrm{Cont}(A, B) \lhd_{(c)} A \to B.$$

Proof. Let Graph and Funct be as in Remark 2.1. Set $\overline{\mathrm{Graph}} = \mathrm{Graph} \circ \mathrm{Ext}$. Then, $(\overline{\mathrm{Graph}}, \mathrm{Funct})$ is the required continuous (and computable) embedding, by Remarks 2.1 and Lemma 2.9. □

Our main result (Theorem 2.12 below) uses a generalized version of Myhill and Shepherdson's (GMS) theorem. This is stated in [16]; a proof may be found in [17]. The GMS theorem relates effective operators over numbered sets to continuous and

computable functions in the category of domains. Recall that F from $\{c_n\}_{n\in\omega}$ to $\{d_n\}_{n\in\omega}$ is an *effective operator* iff, for some recursive function f, $F(c_n) = d_{f(n)}$. It is easy to give some 'natural' (Gödel) numbering to the computable part X_c of an effectively given domain (see the principal computable enumeration in [16], [17], or [10, 11], where GMS is discussed in domains with a suitable notion of application). Of course, in the interesting cases, this (Gödel) numbering is not one-one.

Theorem 2.11 (GMS). *Let* $(X, X_0, \nu, \leqslant)$ *and* $(Y, Y_0, \mu, \leqslant)$ *be effectively given domains. Let* $X_c = \{c_n\}_{n\in\omega}$ *and* $Y_c = \{d_n\}_{n\in\omega}$ *as above and let* f *map* X *to* Y. *Then* f *is (or induces) an effective operator from* X_c *to* Y_c *iff* $f \in \mathrm{Cont}(X, Y)_c$.

Consider now the category of domains and assume that, for any $\phi \in \mathrm{AT}$, $(X^\phi, X_0^\phi, \leqslant)$ is a domain. Then set, for $\sigma, \tau \in T$, $X^{\sigma\to\tau} = \mathrm{Cont}(X^\sigma, X^\tau)$ and $X^{\sigma\times\tau} = X^\sigma \times X^\tau$, as usual.

Theorem 2.12. *Let* $(X^\phi, X_0^\phi, \nu, \leqslant)$ *be an effectively given domain, for all* $\phi \in \mathrm{AT}$. *Assume that, for all* $\phi \in \mathrm{AT}$, $A^\phi \subseteq P\omega$ *is closed, effective and satisfies* $X^\phi \lhd_c A^\phi$. *Then*

$$\forall \sigma \in T, \quad X^\sigma \lhd_c A^\sigma.$$

(As for the definition of A^σ, see Definition 2.5.)

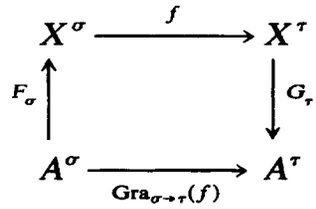
Proof (by induction). Let $X^\sigma \lhd_c A^\sigma$, via (G_σ, F_σ), and $X^\tau \lhd_c A^\tau$, via (G_τ, F_τ). We first prove

$$X^{\sigma\to\tau} = \mathrm{Cont}(X^\sigma, X^\tau) \lhd_c A^\sigma \to A^\tau = A^{\sigma\to\tau}. \tag{1}$$

Define $\mathrm{Gra}_{\sigma\to\tau}$ from $X^{\sigma\to\tau}$ to $\mathrm{Cont}(A^\sigma, A^\tau)$ and $\mathrm{Fun}_{\sigma\to\tau}$ from $\mathrm{Cont}(A^\sigma, A^\tau)$ to $X^{\sigma\to\tau}$ by

$$\mathrm{Gra}_{\sigma\to\tau} = \lambda x. G_\tau \circ x \circ F_\sigma, \qquad \mathrm{Fun}_{\sigma\to\tau} = \lambda x. F_\tau \circ x \circ G_\sigma.$$

(The following diagram visualizes the definition:



A similar diagram may be given for $\mathrm{Fun}_{\sigma\to\tau}$.)

In the category of effectively given domains, the composition operator "$\circ$" is continuous and computable (see [43]). Let $X_c^{\sigma\to\tau} = \{c_n\}_{n\in\omega}$ and $\mathrm{Cont}(A^\sigma, A^\tau)_c = \{d_n\}_{n\in\omega}$. By assumption, G_τ and F_σ are continuous and computable. Then, by the GMS theorem for "$\circ$", for some recursive function f,

$$\mathrm{Gra}_{\sigma\to\tau}(c_n) = G_\tau \circ c_n \circ F_\sigma = d_{f(n)}$$

(of course, f depends uniformly effectively on (the indices for) G_τ and F_σ). By GMS again, $\mathrm{Gra}_{\sigma\to\tau} \in \mathrm{Cont}(X^{\sigma\to\tau}, \mathrm{Cont}(A^\sigma, A^\tau))_c$. Similarly for $\mathrm{Fun}_{\sigma\to\tau}$.

Now, set $G_{\sigma\to\tau} = \overline{\mathrm{Graph}} \circ \mathrm{Gra}_{\sigma\to\tau}$, where $\overline{\mathrm{Graph}} = \mathrm{Graph} \circ \mathrm{Ext}$, and

$$F_{\sigma\to\tau} = \mathrm{Fun}_{\sigma\to\tau} \circ \mathrm{Funct}.$$

For $f \in \mathrm{Cont}(P\omega, P\omega)$ such that $\mathrm{range}(f \restriction A^\sigma) \subseteq A^\tau$, set

$$\mathrm{Fun}_{\sigma\to\tau}(f) = \mathrm{Fun}_{\sigma\to\tau}(f \restriction A^\sigma). \tag{2}$$

Compute then, for $f \in X^{\sigma\to\tau}$,

$$\begin{aligned} F_{\sigma\to\tau} \circ G_{\sigma\to\tau}(f) &= \mathrm{Fun}_{\sigma\to\tau} \circ \mathrm{Funct} \circ \overline{\mathrm{Graph}} \circ \mathrm{Gra}_{\sigma\to\tau}(f) \\ &= \mathrm{Fun}_{\sigma\to\tau} \circ \mathrm{Gra}_{\sigma\to\tau}(f) \quad \text{by Remark 2.1(i) and (2)} \\ &= F_\tau \circ G_\tau \circ f \circ F_\sigma \circ G_\sigma \\ &= f. \end{aligned}$$

$G_{\sigma\to\tau}$ and $F_{\sigma\to\tau}$ are clearly continuous and computable. □

In view of Theorem 1.5, any effectively given domain is (canonically) a computable retraction of some closed and effective subset of $P\omega$. In the sequel, though, we shall also use another natural embedding.

Note now that computable maps, such as G_σ and F_σ, take computable objects to computable objects. Then, by Theorem 2.12, one may understand computable functionals in an abstract structure simply by r.e. sets and "$\cdot$" over $P\omega$. For example, in the notation of Theorem 2.12, let $f \in \mathrm{Cont}(X^\sigma, X^\tau)_c$. By the definition, this means that the 'ideal below' f has an r.e. set of indices. The ideal below, though, as a set of finite sups of step functions (and their indices), is not an immediate notion to handle. By Theorem 2.12, f is characterized by $G_{\sigma\to\tau}(f) \in A^\sigma \to A^\tau \cap \mathrm{RE}$, i.e., by an r.e. set which takes, by "$\cdot$", A^σ into A^τ.

3. Recursion theory

This section is devoted to an application of the previous results to the number-theoretic hierarchy of functionals. In particular, we relate a type structure in $P\omega$ to the continuous and computable functionals, in any finite type, over the flat cpo $\omega^\perp$ of integers, with the Scott topology. Ershov (see [16] for an account and references therein) characterized by this the Kleene–Kreisel countable (continuous) functionals [26, 27] and the Hereditarely Effective Operations (HEO; see [46]).

Let $\omega^\perp$ be as above and embed ω with the discrete topology. Then $E = \mathrm{Cont}(\omega, \omega^\perp)$ are the partial functions from ω to ω. E is an effectively given domain and is isomorphic to

$$P = \{\{\langle n, m\rangle \mid m = f(n)\} \mid f \in E\} \subseteq P\omega.$$

Let T be a collection of function type symbols, as in Definition 2.3, with only one atomic type, (1), say. Set then, for $\sigma, \tau \in T$, $E^{(1)} = E$, $E^{\sigma\to\tau} = \mathrm{Cont}(E^\sigma, E^\tau)$, and $P^{(1)} = P$, $P^{\sigma\to\tau} = P^\sigma \to P^\tau$.

Theorem 3.1. $\forall \sigma \in T$, $E^\sigma \lhd_c P^\sigma$, *via* (G_σ, F_σ).

Proof. In view of the isomorphism between $E^{(1)}$ and $P^{(1)}$, via $(G_{(1)}, F_{(1)})$ say, the result follows from Theorem 2.12. □

As pointed out after Theorem 2.12, one then has, for each $\sigma \in T$, $f \in E^\sigma_c$ iff $G_\sigma(f) \in P^\sigma_c$, where by Lemma 2.2, P^σ_c is well defined and $P^\sigma_c = P^\sigma \cap \mathrm{RE}$.

Now, set $C^{(1)} = \mathrm{Cont}(\omega^\perp, \omega^\perp)$, $C^{\sigma\to\tau} = \mathrm{Cont}(C^\sigma, C^\tau)$. By results in [16], $\mathbb{C} = \{C^\sigma \mid \sigma \in T\}$ modulo some equivalence relations (see below) gives a model of the Kleene–Kreisel countable functionals.

Proposition 3.2. $\forall \sigma \in T$, $E^\sigma \lhd_c C^\sigma$.

Proof. For $f \in E^{(1)}$ and $x \in \omega^\perp$, define

$$i_{(1)}(f)(x) = \textbf{if } x \neq \perp \textbf{ then } f(x) \textbf{ else } \perp.$$

For $g \in C^{(1)}$, set $j_{(1)}(g) = g \restriction \omega$. As for "$\to$"-types, set

$$i_{\sigma\to\tau}(f) = i_\tau \circ f \circ j_\sigma \quad \text{and} \quad j_{\sigma\to\tau}(g) = j_\tau \circ g \circ i_\sigma.$$

The rest of the argument goes similarly as in the proof of Theorem 2.12. □

Both the E^σs and the C^σs are hierarchies of partial functionals. This is not so for the countable functionals and the HEO which are total maps. Following [16], define

$$\mathrm{CO}^{(1)} = \{f \in E^{(1)} \mid \forall n \in \omega, f(n) \neq \perp\},$$

$$\mathrm{CO}^{\sigma\to\tau} = \{f \in E^{\sigma\to\tau} \mid \forall g \in \mathrm{CO}^\sigma, f(g) \in \mathrm{CO}^\tau\}$$

and

$$\mathrm{HE}^{(1)} = \{f \in E^{(1)}_c \mid \forall n \in \omega, f(n) \neq \perp\},$$

$$\mathrm{HE}^{\sigma\to\tau} = \{f \in E^{\sigma\to\tau}_c \mid \forall g \in \mathrm{HE}^\sigma, f(g) \in \mathrm{HE}^\tau\}.$$

We can mimic this within $P\omega$. That is, set

$$P^{(1)}_{\mathrm{CO}} = \{\{\langle n, m\rangle \mid m = f(n)\} \mid f : \omega \to \omega \text{ total}\},$$

$$P^{\sigma\to\tau}_{\mathrm{CO}} = (P^\sigma_{\mathrm{CO}} \to P^\tau_{\mathrm{CO}}) \cap P^{\sigma\to\tau},$$

and

$$P^{(1)}_{\mathrm{HE}} = \{\{\langle n, m\rangle \mid m = f(n)\} \mid f : \omega \to \omega \text{ total } \textbf{and} \text{ computable}\},$$

$$P^{\sigma\to\tau}_{\mathrm{HE}} = (P^{\sigma}_{\mathrm{HE}} \to P^{\tau}_{\mathrm{HE}}) \cap P^{\sigma\to\tau}_{\mathrm{c}},$$

It should be clear that the elements of $\mathrm{CO}^{(1)}$ are total maps and the elements in $\mathrm{CO}^{\sigma\to\tau}$ take the total maps in CO^{σ} to CO^{τ}. Similarly, $f \in \mathrm{HE}^{\sigma\to\tau}$ are total and computable functionals taking HE^{σ} to HE^{τ}. By this they are hereditarily total (and computable). Similarly, one may view at P^{σ}_{CO} and P^{σ}_{HE} within $\langle P\omega, \cdot\rangle$.

It is easy to show that $\forall \sigma \in T$, $P^{\sigma}_{\mathrm{CO}} \neq \emptyset$ and $P^{\sigma}_{\mathrm{HE}} \neq \emptyset$.

Theorem 3.3. *For all $\sigma \in T$ one has*

(i) $\mathrm{CO}^{\sigma} \lhd_{\mathrm{c}} P^{\sigma}_{\mathrm{CO}}$ *via* (G_σ, F_σ),

(ii) $\mathrm{HE}^{\sigma} \lhd_{\mathrm{c}} P^{\sigma}_{\mathrm{HE}}$ *via* (G_σ, F_σ),

(iii) $\mathrm{HE}^{\sigma} \lhd_{\mathrm{c}} P^{\sigma}_{\mathrm{CO}} \cap \mathrm{RE}$ *via* (G_σ, F_σ)

(*where* (G_σ, F_σ) *is as in Proposition* 3.1).

Proof. By the inductive definition in Theorem 2.12, recall first that, for $f \in E^{\sigma\to\tau}$ and $a \in P^{\sigma}$,

$$G_{\sigma\to\tau}(f) \cdot a = G_\tau(f(F_\sigma(a))) \tag{3}$$

and, for $b \in P^{\sigma\to\tau}$ and $g \in E^{\sigma}$,

$$F_{\sigma\to\tau}(b)(g) = F_\tau(b \cdot G_\sigma(g)), \tag{4}$$

$$P^{(1)}_{\mathrm{CO}} \cong \mathrm{CO}^{(1)}, \quad \text{via } G_{(1)} \restriction \mathrm{CO}^{(1)} \text{ and } F_{(1)} \restriction P^{(1)}_{\mathrm{CO}}. \tag{5}$$

As for the nontrivial inductive step we need to show that, for $f \in \mathrm{CO}^{\rho\to\tau}$ and $a \in P^{\rho}_{\mathrm{CO}}$, $G_{\rho\to\tau}(f) \cdot a \in P^{\tau}_{\mathrm{CO}}$. This immediately follows from (3) and the inductive hypothesis.

Then use (4) and the inductive hypothesis to prove that, for $b \in P^{\rho\to\tau}_{\mathrm{CO}}$ and $g \in \mathrm{CO}^{\rho}$, $F_{\rho\to\tau}(b)(g) \in \mathrm{CO}^{\tau}$. Parts (ii) and (iii) follow similarly, by using $f \in E^{\sigma}_{c}$ iff $G_\sigma(f) \in P^{\sigma}_{c}$ $(= P^{\sigma} \cap \mathrm{RE})$. □

Notice that $P^{(1)}_{\mathrm{HE}} \subseteq P^{(1)}_{\mathrm{CO}}$. At higher types, the hierarchies are no longer comparable by set inclusion. Theorem 3.3(ii), (iii), though, gives $G_\sigma(\mathrm{HE}^{\sigma}) \subseteq P^{\sigma}_{\mathrm{HE}} \cap P^{\sigma}_{\mathrm{CO}}$.

By results in [16] and [31], the CO^{σ}'s and the HE^{σ}'s, modulo an equivalence relation, give the countable functionals and the HEO, that is widely studied classes of recursion-theoretic functionals.

The whole story then amounts to representing functionals within $\langle P\omega, \cdot\rangle$. Of course, if one says that $a \in P^{\sigma}$ *represents* $f \in E^{\sigma}$ iff $F_\sigma(a) = f$, then each $f \in E^{\sigma}$ has many representatives. They are nicely behaved, though.

Definition 3.4. Let $\sim_{(1)}$ be the identity relation over $P^{(1)}$; for $a, b \in P^{\sigma\to\tau}$, define

$$a \sim_{\sigma\to\tau} b \text{ iff } \forall c \in P^{\sigma},\ ac \sim_\tau bc.$$

The meaning of the "$\sim_\sigma$" equivalence relations should be clear: $a \sim_{(1)\to(1)} b$ iff a and b are extensionally equivalent on P, and so on.

Theorem 3.5. *Let $a, b \in P^\sigma$. Then one has*

(i) $[a]_\sigma = \{c \in P^\sigma \mid c \sim_\sigma a\}$ *is a complete lattice* (*w.r.t.* $\subseteq$).

(ii) $a \sim_\sigma b \Rightarrow F_\sigma(a) = F_\sigma(b)$.

Proof. (i) $[a]_{(1)} = \{a\}$. We only check that, for $X \subseteq [a]_{\sigma\to\tau}$, $\bigcup X \in [a]_{\sigma\to\tau}$. By the continuity of "$\cdot$", $(\bigcup X)d = \bigcup_{x\in X}(xd)$. Let $d \in P^\sigma$; then $\forall x \in X$, $xd \in [ad]_\tau$ and $\bigcup_{x\in X}(xd) \sim_\tau ad$ by the induction hypothesis.

(ii) Notice first that

$$\begin{aligned} &F_{\sigma\to\tau}(a) = F_{\sigma\to\tau}(b) \\ &\quad\Leftrightarrow \forall f \in E^\sigma,\ F_{\sigma\to\tau}(a)(f) = F_{\sigma\to\tau}(b)(f) \\ &\quad\Leftrightarrow \forall f \in E^\sigma,\ F_\tau(a \cdot G_\sigma(f)) = F_\tau(b \cdot G_\sigma(f)). \end{aligned} \tag{6}$$

Assume now that $a \sim_{\sigma\to\tau} b$. Then, $a \cdot G_\sigma(f) \sim_\tau b \cdot G_\sigma(f)$ and (6) holds by the induction hypothesis. □

4. A completeness theorem for formal type assignment

Type symbols may be regarded as constraints in a language. In Logic, they allow full expressiveness by avoiding paradoxes. In Computer Science, a consistent type discipline prevents runtime errors in programming. In particular, if x is given type σ (*notation*: σx), then f may be formally applied to x iff f has type $\sigma \to \tau$ (i.e., $\sigma \to \tau f$), for some type τ.

This has been formalized by Curry and Feys [13] for typed λ-calculus, which is the epitome for a well-understood functional theory of types. Pioneering work may be found in [20].

Definition 4.1. Let T be the collection of function types (see Definition 2.3).

(i) A *basis* for type assignment is a set $B = \{\sigma x, \tau y, \ldots\}$ where $\sigma, \tau, \ldots \in T$ and $x \not\equiv y, \ldots$ are variables.

(ii) *Curry type assignment* is defined by the following natural deduction system:

$$(\to\mathrm{I})\ \frac{\begin{array}{c}[\sigma x]\\ \vdots \\ \tau M\end{array}}{\sigma \to \tau\ \lambda x.M}\ ^{1} \qquad (\to\mathrm{E})\ \frac{\sigma \to \tau M\ \ \sigma N}{\tau(MN)} \qquad (\mathrm{Eq}_\beta)\ \frac{\sigma M\ \ \lambda\beta \vdash M = N}{\sigma N}$$

(iii) If σM is derivable from a basis B, then write $B \vdash \sigma M$.

[1] If x is not free in assumptions on which τM depends.

A semantics for Curry's types may be given in several ways. A simple one interprets types as subsets of a model for (type-free) λ-calculus, see [7, "open problem II-4"].

Definition 4.2. Let $\mathcal{M} = \langle D, \cdot, [\![\ ]\!] \rangle$ be a λ-model (see [22, 34, 2], say).

(i) For $\xi : \mathrm{Var} \to D$, $[\![M]\!]_\xi$ is the interpretation of the λ-term M in $\mathcal{M}$ via ξ.

(ii) Let $V : At \to PD = \{A \mid A \subseteq D\}$. Then the interpretation of $\sigma \in T_F$ in $\mathcal{M}$ via V is defined by:

(1) $[\![\phi_i]\!]_V = V(\phi_i)$, for $\phi_i \in At$,

(2) $[\![\sigma \to \tau]\!]_V = \{d \in D \mid \forall e \in [\![\sigma]\!]_V,\ d \cdot e \in [\![\tau]\!]_V\}$.

(iii) $\mathcal{M}, \xi, V \models \sigma M$ iff $[\![M]\!]_\xi \in [\![\sigma]\!]_V$,

$\mathcal{M}, \xi, V \models B$ iff $\mathcal{M}, \xi, V \models \sigma x$, for all $\sigma x \in B$,

$B \models \sigma M$ iff for all $\mathcal{M}, \xi, V$, $\mathcal{M}, \xi, V \models B \Rightarrow \mathcal{M}, \xi, V \models \sigma M$.

It is easy to prove the soundness of this semantics, i.e.,

$$B \vdash \sigma M \Rightarrow B \models \sigma M. \tag{7}$$

The completeness, i.e., the reverse implication, has been first proved in [4, 21], by using models of type symbols and term models. A purely semantic proof is given in [9], by using a special purpose construction over the λ-model $\langle P\omega, \cdot, [\![\ ]\!] \rangle$. In this section, by borrowing from [9, 12], we show that the completeness result may be given over a 'meaningful' model. Namely, the type structure $\{P^\sigma\}_{\sigma \in T_F}$ within $P\omega$. By the previous results, this relates, via $P\omega$, the recursion-theoretic type structure over the partial function on ω to the formal assignment of types to λ-terms.

Summary. The completeness proof goes as follows. We first choose $V_0 : At \to PP\omega$ as $\forall \phi_i \in At$, $V_0(\phi_i) = P^{(1)}$ $(= P)$; thus, $\forall \sigma \in T$, $[\![\sigma]\!]_{V_0} = P^\sigma$. Then we construct, for any given basis B, a collection R_B of environments such that, for all $\xi \in R_B$, $P\omega, \xi, V_0 \models B$ (Definition 4.5). Finally, we prove that if, for any $\xi \in R_B$, $P\omega, \xi, V_0 \models \sigma M$, i.e., $[\![M]\!]_\xi \in P^\sigma$, then one has $B \vdash \sigma M$ (Theorem 4.8; completeness). Remark 4.9 proves the completeness of the F-semantics over the $\{P^\sigma\}_{\sigma \in T}$ model, i.e., by taking a unique representative in $P^{\sigma \to \tau}$ for each function in $P^\sigma \to P^\tau$ (hereditarily over types).

Question. The semantics of typed λ-terms may be given over a type structure of domains, e.g., $\{E^\sigma\}_{\sigma \in T}$ in Section 3, in a natural way (see [39], say). One may also view at *typed* λ-terms as a type σ, a term M together with a deduction $B \vdash \sigma M$ and, then, using Theorem 2.12 and (7), to interpret any (closed) σM by $F_\sigma([\![M]\!]_\xi) \in E^\sigma_c$, say (it is easy to see that any closed λ-term is interpreted by an r.e. set in $P\omega$). It would be interesting to characterize those higher type computable functionals which are taken (by G_σ, see Theorems 2.12 and 3.1) to the λ-definable elements (of P^σ). (Of course, by the language LAMBDA [42] and by Theorem 2.12 one could recover all of them.)

Note, finally, that what is also missing in the literature is a completeness theorem over a type structure, of recursion-theoretic interest, where terms are interpreted by functions in extenso. Remark 4.9 goes very close to this by proving the completeness of the F-semantics over the given type structure within $P\omega$.

Preliminaries (and notation). $\forall m, n > 1$, $n, m < \langle n, m\rangle$. $e_0 = \emptyset$. $\lambda x.f(x) := \mathrm{Graph}(f)$ $(= \{\langle n, m\rangle \mid m \in f(e_n)\}$; see Remark 2.1) and $\lambda x.f(x)$ is *saturated* (i.e., $\langle n, m\rangle \in \lambda x.f(x)$ and $e_n \subseteq e_p$ imply $\langle p, m\rangle \in \lambda x.f(x)$).

Note that, for $A, B \subseteq P\omega$ and $a \in P\omega$, $a \in A \to B$ implies $a \subseteq \lambda x.ax \in A \to B$.

By the intended choice of $\langle\ ,\ \rangle$ and $\{e_n\}_{n\in\omega}$, a closed λ-term is interpreted by $\emptyset \in P\omega$ iff it is unsolvable (see [3]).

Finally, set $A \cdot B = \{ab \mid a \in A$ **and** $b \in B\}$.

Lemma 4.3. (i) $\forall \sigma \in T$, $\forall e_n \in P^\sigma$, $\exists a \in P\omega$ *infinite*, $e_n \subseteq a \in P^\sigma$ ***and*** $\exists m \in \omega$, $e_n \subseteq e_m \notin P^\sigma$.

(ii) $\forall \sigma, \forall n$ $(\sigma \neq (1)$ ***and*** $e_n \subseteq P^{(1)} \cap P^\sigma \Rightarrow \exists m,\ e_n \subseteq e_m \in P^{(1)} \setminus P^\sigma)$.

(iii) $\forall \sigma, \exists m_1, \exists m_2, \{m_1, m_2\} \notin P^\sigma$.

Proof. (i) $\sigma = (1)$: trivial. $\sigma = \rho \to \tau$: take $a = \lambda x.e_n x$. Moreover, observe that P^σ is closed, by Lemma 2.2 for P is closed, and that $\omega \notin P^\sigma$, by Lemma 2.7.

(ii) Let $\sigma = \rho \to \tau$. Recall that $e_n \in P^{\rho\to\tau}$ iff $\forall J \subseteq e_n$.

$$\bigcup_{\langle p,q\rangle\in J} e_p \in P^\rho \Rightarrow \bigcup_{\langle p,q\rangle\in J} \{q\} \in P^\tau. \tag{8}$$

Let J be maximal such that (8) holds. Define

$$e_v = \bigcup_{\langle p,q\rangle\in J} e_p \quad \text{and} \quad e_r = \bigcup_{\langle p,q\rangle\in J} \{q\}.$$

Then, take $e_t \notin P^\tau$, $e_r \subseteq e_t$, by Lemma 4.3(i), and construct a chain $e_v \subsetneq e_{v_1} \subsetneq e_{v_2} \subsetneq \cdots \subseteq a \in P^\rho$, by Lemma 4.3(i) again. For $e_t = e_r \cup \{n_1, \ldots, n_h\}$, take $e_m = e_n \cup \{\langle v_i, n_i\rangle \mid i = 1, \ldots, h\}$.

Now, note that $\forall \langle p, q\rangle \in e_n, \forall v_i$ (in the chain), $v_i \neq p$, for J is maximal. Then, compute

$$\begin{aligned} e_m e_{v_h} &= e_n e_{v_h} \cup (\{\langle v_i, n_i\rangle \mid i = 1, \ldots, h\} \cdot e_{v_h}) \\ &= e_r \cup \{n_1, \ldots, n_h\} = e_t \notin P^\tau. \end{aligned}$$

(iii) $\sigma = (1)$: trivial. $\sigma = \mu \to \nu$: take $\langle 0, m_1\rangle$ and $\langle 0, m_2\rangle$ for $\{m_1, m_2\} \notin P^\nu$ $(m_1 \neq m_2)$. □

Theorem 4.4. $\forall \sigma, \tau$, $P^\sigma \subseteq P^\tau \Rightarrow \sigma = \tau$.

Proof. $\sigma = (1)$, $\tau = (1)$: trivial.

$\sigma = (1)$, $\tau = \delta \to \gamma$: by Lemma 4.3(ii).

$\sigma = \delta \to \gamma$, $\tau = (1)$: $P^\sigma \not\subseteq P^{(1)}$, for no element of $P^{(1)}$ is saturated.

$\sigma = \delta \to \gamma$, $\tau = \alpha \to \beta$: by induction and Proposition 2.8. □

Definition 4.5. (i) $\Delta_\sigma = \lambda x.f_\sigma(x)$, where $f_\sigma(x) =$ "**if** $x \in P^\sigma$ **then** $\lambda x.x$ **else** ω".

(ii) $T^{(1)} = P^{(1)}$, $T^{\sigma\to\tau} = \{\lambda x.\Delta_\sigma xy \mid y \in T^\tau\}$.

(iii) Let B be a basis (see Definition 4.2). Then, set

$$R_B = \{\xi_t : \mathrm{Var} \to P\omega / \xi_t(x) = \text{"}\mathbf{if}\ \sigma x \in B\ \mathbf{then}\ t\ \mathbf{else}\ \omega\text{"},\ t \in T^\sigma\}.$$

(iv) Let $R \subseteq \mathrm{Env}$ and M be a λ-term. Then, set $[\![M]\!]_R = \{[\![M]\!]_\xi \mid \xi \in R\}$.

(Note that Δ_σ in (i) is well defined, for P^σ is closed and, hence, f_σ is continuous.)

Lemma 4.6. (i) $\forall\sigma,\ T^\sigma \subseteq P^\sigma$.

(ii) $\forall\sigma, \tau,\ T^\sigma \subseteq P^\tau \Rightarrow \sigma = \tau$.

Proof. (i) easily follows by induction.

(ii) $\sigma = (1)$: by Theorem 4.4.

$\sigma = \alpha \to \beta$, $\tau = (1)$: trivial.

$\sigma = \alpha \to \beta$, $\tau = \gamma \to \delta$:

$$T^\sigma \subseteq P^\tau \Leftrightarrow \forall a \in P^\gamma, \forall b \in T^\beta, \Delta_\alpha ab \in P^\delta$$
$$\Leftrightarrow P^\gamma \subseteq P^\alpha \ \mathbf{and}\ T^\beta \subseteq P^\delta.$$

Then use the inductive hypothesis and Theorem 4.4. □

Let now B be a basis and $\sigma z = \sigma_1 \to (\sigma_2 \to \cdots \to (\sigma_k \to (1)) \ldots) z \in B$. Then, by definition,

$$\forall \xi \in R_B,\quad \xi(z) = \lambda x_1.\Delta_{\sigma_1} x_1(\lambda x_2.\Delta_{\sigma_2} x_2(\ldots(\lambda x_k.\Delta_{\sigma_k} x_k c)\ldots), \text{ for some } c \in P^{(1)}.$$

(*Convention*: $\xi(z) = a_c^\sigma$; $[m] = n$ if $\{m\} = e_n$.)

Lemma 4.7. *Let N be a term in normal form and B a basis. Then there exists a $\xi \in R_B$ such that*:

(1) *for any free variable z in N, if $(1)z \in B$, then $\xi(z)$ is finite; otherwise, if $\sigma \to \tau\, z \in B$ and $\xi(z) = a_c^{\sigma\to\tau}$, then c is finite*;

(2) $[\![N]\!]_\xi \in P\omega$ *has at least two elements,* n_1, n_2, *say, such that* $n_1, n_2 > \max\{k \mid \exists z \in \mathrm{Var}, \exists m\ ((1)z \in B$ ***and*** $\langle k, m\rangle \in \xi(z))$ ***or*** $(\sigma \to \tau\, z \in B$ ***and*** $\xi(z) = a_c^{\sigma\to\tau}$ ***and*** $\langle k, m\rangle \in c)\}$.

Proof (by induction on the structure of N).

Case ($N \equiv x$). Let $c = \{\langle 1, 1\rangle, \langle 2, 2\rangle\}$ and set

$$\xi(z) = \begin{cases} c & \text{if } (1)z \in B, \\ a_c^\sigma & \text{if } \sigma z \in B, \text{for } \sigma \neq (1), \\ \omega & \text{otherwise.} \end{cases}$$

Clearly, $\xi \in R_B$, and (1) and (2) hold.

Case ($N \equiv \lambda y.N'$). By definition, $[\![\lambda y.N]\!]_\xi = \lambda d.[\![N']\!]_{\xi[d/y]}$ and y is not free in N. By induction, let $\xi \in R_B$ satisfies (1) for N'; then, clearly, ξ satisfies (1) for N also. By induction, again, let $n_1, n_2 \in [\![N']\!]_\xi$ satisfy (2) and take $e_m \subseteq \xi(y)$ s.t. $n_1, n_2 \in [\![N']\!]_{\xi[e_m/y]}$. Then, $\langle m, n_1\rangle, \langle m, n_2\rangle \in [\![N]\!]_\xi$ satisfy (2).

Case ($N \equiv xN_1 \dots N_m$). (i) $(1)x \in B$. Let n_1, n_2, and ξ_1 satisfy, by induction, (1) and (2) for N_1. Then, set $\xi(y) = \xi_1(y)$ for $y \neq x$ and

$$\xi(x) = \xi_1(x) \cup \{\langle [n_1], \underbrace{\langle 0, \langle 0, \dots, \langle 0}_{m-1}, [n_1]+1\rangle \cdots\rangle,$$

$$\langle [n_2], \underbrace{\langle 0, \langle 0, \dots, \langle 0}_{m-1}, [n_2]+1\rangle \cdots\rangle\}.$$

$[n_1]+1, [n_2]+2$, and ξ easily satisfy (1) and (2) for N.

(ii) $\sigma_1 \to (\sigma_2 \to \cdots \to (\sigma_k \to (1)) \dots)x \in B$, with $k \geq 1$.

Subcase ($m \leq k$). Assume, by induction, that n_1, n_2, and ξ_m satisfy (1) and (2) for N_m, with $\xi_m(x) = a_c$. Let $n = n_1 + n_2$, $c_0 = \{\langle n, n\rangle, \langle n+1, n+1\rangle\}$ and set $\xi(x) = a_{c \cup c_0}$ and $\xi(y) = \xi_m(y)$, for $y \neq x$. Then $\langle n, n\rangle$, $\langle n+1, n+1\rangle$, and ξ do the job for N.

Subcase ($k < m$). Assume that n_1, n_2, and ξ_k satisfy (1) and (2) for N_{k+1}, with $\xi_k(x) = a_c$. Then, set

$$c_0 = \{\langle [n_1], \underbrace{\langle 0, \langle 0, \dots, \langle 0}_{m-k-2}, [n_1]+1\rangle \cdots\rangle,$$

$$\langle [n_2], \underbrace{\langle 0, \langle 0, \dots, \langle 0}_{m-k-2}, [n_2]+1\rangle \cdots\rangle\},$$

$\xi(x) = a_{c \cup c_0}$ and $\xi(y) = \xi_k(y)$, for $y \neq k$. The rest is analogous to the subcase $m \leq k$.

(iii) x does not appear in B. Set $\xi(x) = \omega$. (Note that in $\max\{\cdots\}$ given in (2) above, also when looking at infinitely many $z \in \mathrm{Var}$, one actually deals with finitely many finite sets $\xi(z)$ or c, depending on the length of N, by the construction of ξ.) □

Theorem 4.8. *Let N be a term possessing normal form and B a basis. Then*

$$[\![N]\!]_{R_B} \subseteq P^\sigma \Rightarrow B \vdash \sigma N.$$

Proof. As $P\omega$ is a model, we may assume that N *is* in normal form.

Case ($N \equiv x$). $[\![x]\!]_{R_B} \subseteq P^\sigma \Rightarrow \exists \tau \in T_F$, $\tau x \in B$, for $\omega \notin P^\sigma$. Then, $T^\tau \subseteq P^\sigma$, by definition of R_B, and $\tau = \sigma$, by Lemma 4.6.

Case ($N \equiv \lambda x.N'$). Clearly, $\sigma \neq (1)$, for no element of $P^{(1)}$ is saturated. Now, for $\mu x \in B$, if any, $[\![\lambda x.N']\!]_{R_B} \subseteq P^{\gamma \to \delta} \Rightarrow [\![\lambda x.N']\!]_{R_B} \cdot T^\gamma \subseteq P^\delta$, by $T^\gamma \subseteq P^\delta$, this implies $[\![N']\!]_{R_{B'}} \subseteq P^\delta$, where $B' = B \setminus \{\mu x\} \cup \{\gamma x\}$, by $(\xi \in R_{B'} \Rightarrow \xi(x) \in T^\gamma)$. Thus, by induction, $B \setminus \{\mu x\} \cup \{\gamma x\} \vdash \delta N'$, hence, $B \vdash \gamma \to \delta \lambda x.N'$.

Case ($N \equiv xN_1 \dots N_m$). $[\![N]\!]_{R_B} \subseteq P^\sigma \Rightarrow \exists \tau \in T_F$, $\tau x \in B$, for $\omega \notin P^\sigma$. Suppose first that $\tau = (1)$, i.e., suppose that $(1)x \in B$. Let n_1, n_2, and $\xi_1 \in R_B$ be given for N_1 as in Lemma 4.7. Fix $\{p_1, p_2\} \notin P^\sigma$ (see Lemma 4.3(iii)) and set

$$\xi(x) = \xi_1(x) \cup \{\langle [n_1], \langle 0, \langle 0, \dots, \langle 0, p_1\rangle \cdots\rangle, \langle [n_2], \langle 0, \langle 0, \dots, \langle 0, p_2\rangle \cdots\rangle\},$$

$\xi(y)=\xi_1(y)$, for $y\not\equiv x$. Note that $\xi(x)\in P^{(1)}$ by Lemma 4.7(1), (2). Then, $[\![xN_1\dots N_m]\!]_\xi=[\![N]\!]_{\xi_1}\cup\{p_1,p_2\}$, for $\{n_1\},\{n_2\}\subseteq[\![N_1]\!]_{\xi_1}$ by easy computation; moreover, $[\![N]\!]_{\xi_1}\cup\{p_1,p_2\}\notin P^\sigma$, since P^σ is a closed subset. Thus, $\tau\neq(1)$, that is for some $\mu,\nu\in T_F$ one has $\tau=\mu\to\nu$. Then, compute

$$[\![xN_1\dots N_m]\!]_{R_B}=T^{\mu\to\nu}[\![N_1\dots N_m]\!]_{R_B}=\Delta_\mu[\![N_1]\!]_{R_B}.T^\nu\cdot[\![N_2\dots N_m]\!]_{R_B}\ (:=L).$$

Thus, $[\![N_1]\!]_{R_B}\subseteq P^\mu$, for $\omega\notin[\![N]\!]_{R_B}\subseteq P^\sigma$, and, hence

$$B\vdash\mu N_1,\quad\text{by induction.}\tag{9}$$

Now note that, for L as above, one has $L=T^\nu[\![N_2\dots N_m]\!]_{R_B}=[\![yN_2\dots N_m]\!]_{R_{B'}}$, for $B'=B\cup\{\nu y\}$) and y not occurring in B. Thus, by the induction hypothesis and by $L=[\![N]\!]_{R_B}\subseteq P^\sigma$, $B\cup\{\nu y\}\vdash\sigma yN_2\dots N_m$. In particular, by a small abuse of language, one has

$$B\cup\{\nu(xN_1)\}\vdash\sigma xN_1N_2\dots N_m.\tag{10}$$

Observe now that, by $(\mu\to\nu)x\in B$, (9) and ($\to$E), one has $B\vdash\nu(xN_1)$. Therefore, $B\vdash\sigma xN_1\dots N_m$, by (10). □

As is well known, each typable term, in the sense of Curry, possesses a normal form (see, [23] say).

Remark 4.9 (*the completeness w.r.t. the F-semantics*). There is at least one more way to interpret type assignment over λ-models in such a manner that each representable function has a unique representative. Namely, given $\langle D,\cdot,[\![\ ]\!]\rangle$, define $F=\{[\![\lambda y.xy]\!]_\xi\mid\xi\in\mathrm{Env},x\in\mathrm{Var}\}$. Then, set, for a choice of $V(\phi_i)\subseteq D$,

$$[\![\sigma\to\tau]\!]^F_V=\{d\in F\mid\forall e\in[\![\sigma]\!]^F_V,\ de\in[\![\tau]\!]^F_V\}$$

(the F-(functional)-semantics).

As for the $P\omega$ model, $F=\mathrm{Graph}(\mathrm{Cont}(P\omega,P\omega))$, that is, F is exactly the set of the 'generalized graphs' of the continuous functions, one for each function, of course.

Then, define $V_0(\phi_i)=P^{(1)}$ $(=P)$ and $PF^\sigma=[\![\sigma]\!]^F_{V_0}$, i.e., $PF^{\sigma\to\tau}=\{d\in F\mid\forall e\in PF^\sigma,\ de\in PF^\tau\}$. The point is that Theorem 4.8 holds by writing PF^σ instead of P^σ. We just list the minor changes needed in the claims, when substituting the PF^σ's for the P^σ's:

- Proposition 2.8: easily holds by observing that $\forall\sigma,\exists n,\ \check{e}_n\subseteq\overline{PF}^\sigma$.
- Theorem 4.4: by Proposition 2.8, new version, and a trivial variant of Lemma 4.3.
- Definition 4.5: in the definition of f_σ, just take the topological closure of PF^σ instead of P^σ.
- Lemma 4.6: by Theorem 4.4 and Definition 4.5, new versions.
- Lemma 4.7: all right (with references to the new T^σ and Δ_σ).
- Theorem 4.8: just write PF^σ instead of P^σ.

5. Concluding remarks

(i) As the reader could have noticed, we had to pay a price in order to use $P\omega$ and the simple semantics of types, as subsets of $P\omega$, for the aim of Theorem 2.12. Namely, we had to use (some) closed subset of $P\omega$ and retractions were given between domains and these subsets ($X^\sigma \lhd_c A^\sigma$ in Theorem 2.12), not the whole of $P\omega$. The advantage we had was a very simple characterization of higher type computability by r.e. sets and the well understood Myhill-Shepherdson "$\cdot$" in $P\omega$. As a consequence, we could apply specific properties of the 'hardware' of $P\omega$ and give the completeness result in the last section. Unfortunately, general (category-theoretic, say) tools for the investigation of completeness properties of type assignment to λ-terms do not seem to exist. The situation seems similar to the characterization of true equalities in models of λ-calculus, where minor differences in the hardware of models modify their theories (see [1, 29, 30] for a discussion). Thus, we fully exploited the 'simplicity' of the simple semantics over $P\omega$ and the work done on $P\omega$ as a λ-model. One of the referees suggested two possible directions for future work. Instead of $P\omega$, which is universal for continuous lattices (i.e., $A \lhd P\omega$ iff A is a continuous lattice), consider the universal domain U for bounded complete cpo's in [43]. U may be taken as the effectively given domain of open subsets of the Cantor space 2^ω, except the largest element. Then, for each effectively given domain A, one would have $A \lhd_c U$. Since $\mathrm{Cont}(U, U) \lhd_c U$, via (ψ, ϕ), say, $(U, \cdot)$, with $u \cdot v = \phi(u)(v)$, would allow a similar investigation of higher type computable functionals within U, without going at closed subsets as in Theorem 2.12 and Section 3. Even more interestingly, one could use, instead of U, a universal domain V for SFP objects, which are suitable for the powerdomain construction and, hence, for the semantics of nonsequential extensions of λ-calculus. Of course, a deeper connection of the type structures within $(U, \cdot)$ or $(V, \cdot)$ to formal type assignment would be given by mimicking the work done in Section 4 over $(P\omega, \cdot)$. This would require a further, entirely different, investigation of $(U, \cdot)$ and $(V, \cdot)$ as models for typed and type-free languages (and of their very abstract notion of application "$\cdot$"). Consider, say, that it is possible that each of $(U, \cdot)$ and $(V, \cdot)$ could be turned into several λ-models (following the terminology and the examples in [29], they do not need to be 'lambda-categorical'). Each model would modify the theory and the type assignment. $(P\omega, \cdot)$ *is* lambda-categorical [8] and, thus, the interpretation of λ-terms, used in Section 4, is the only one which is possible. This simplified our work and made it very natural. The other directions suggested could lead to further very interesting achievements, with applications to nonstrictly sequential languages.

(ii) One can easily extend the collection of P^σ's with product types and then consider them as the objects of a cartesian category, with polynomial (or algebraic) functions as morphisms. This category, however, is not cartesian closed: there are too many representatives (points in $P^{\sigma\to\tau}$) for each morphism from P^σ into P^τ. Nevertheless, it can be shown that it enjoys some 'natural' properties; in a

forthcoming paper a general categorical notion will be introduced (an instance of which is the category of P^{σ}'s) discussing its relations with models of typed and type-free λ-calculus.

Acknowledgment

We are greatly indebted to Mario Coppo for this help in the proof of Theorem 4.8 and for several basic remarks on Section 4.

References

[1] J. Baeten and B. Boerboom, Ω can be anything it shouldn't be, *Indag. Math.* **41** (1979) 111–120.
[2] H.P. Barendregt, Lambda calculus and its models, in: G. Lolli, G. Longo and A. Marcja, eds., *Logic Colloquium '82* (North-Holland, Amsterdam, 1984).
[3] H.P. Barendregt, *The Lambda Calculus: Its Syntax, Its Semantics* (North-Holland, Amsterdam, rev. and expanded ed., 1984).
[4] H.P. Barendregt, M. Coppo and M. Dezani, A filter lambda model and the completeness of type assignment, *J. Symb. Logic* **48** (4) (1983) 931–940.
[5] H.P. Barendregt and G. Longo, Equality of lambda-terms in the model $T\omega$, in: R. Hindley and J. Seldin, eds., *To H.B. Curry: Essays on Combinatory Logic Lambda Calculus and Formalism* (Academic Press, New York, 1980) 303–337.
[6] H.P. Barendregt and G. Longo, Recursion theoretic operators and morphisms of numbered sets, *Fundamenta Math.* **CXIX** (1982) 49–62.
[7] C. Böhm, ed., *Lambda-Calculus and Computer Science*, Lecture Notes in Computer Science **37** (Springer, Berlin, 1975).
[8] K. Bruce and G. Longo, A note on combinatory algebras and their expansions, *Theoret. Comput. Sci.* **31** (1984), 31–41.
[9] M. Coppo, Completeness of type assignment in continuous lambda-models, *Theoret. Comput. Sci.*, **29** (1984) 309–324.
[10] M. Coppo, M. Dezani-Ciancaglini, F. Honsell and G. Longo, Extended type structures and filter lambda-models, in: G. Lolli, G. Longo and A. Marcja, eds., *Logic Colloquium 82* (North-Holland, Amsterdam, 1984).
[11] M. Coppo, M. Dezani-Ciancaglini, F. Honsell and G. Longo, Applicative information systems, and recursive domain equations, in: G. Ausiello and M. Protasi, eds., *Conference on Trees in Algebra and Programming*, Lecture Notes in Computer Science **159** (Springer, Berlin, 1983) (revised: *Inform. and Control*, to appear).
[12] M. Coppo and E. Giovannetti, Completeness results for a polymorphic type systems in: G. Ausiello and M. Protasi, eds., *CAAP '83*, Lecture Notes in Computer Science **159** (Springer, Berlin, 1983).
[13] H.B. Curry and R. Feys, *Combinatory Logic* (North-Holland, Amsterdam, 1958).
[14] H. Egli and R. Constable, Computability concepts for programming languages semantics, *Theoret. Comput. Sci.* **2** (1976) 133–145.
[15] Yu.L. Ershov, Computable functionals of finite types, *Algebra and Logic* **11** (4) (1972).
[16] Yu.L. Ershov, Model C of partial continuous functionals, in: R. Gandy and M. Hyland, eds., *Logic Colloquium 76* (North-Holland, Amsterdam, 1977).
[17] P. Giannini and G. Longo, Effectively given domains and lambda-calculus semantics, *Inform. and Control* **62**(1) (1984) 36–63.
[18] K. Gödel, Uber eine bisher noch nicht benützte Erweiterung des finiten Standpunktes, *Dialectica* **12** (1958) 280–287.
[19] P. Henderson, *Functional Programming* (Prentice-Hall, Englewood Cliffs, N.J., 1980).

[20] R. Hindley, The principal type-scheme of an object in Combinatory Logic, *Trans. A.M.S.* **146** (1969) 29-60.
[21] R. Hindley, The completeness theorem for typing lambda-terms, *Theoret. Comput. Sci.* **22** (1983) 1-18.
[22] R. Hindley and G. Longo, Lambda-calculus models and extensionality, *Z. Math. Logik* **26** (1980) 289-310.
[23] R. Hindley and J. Seldin, *Introduction to Combinators and λ-Calculus* (Cambridge Univ. Press. 1966).
[24] M. Hyland, Filter spaces and continuous functionals, *Ann. Math. Logic* **16** (1979) 101-143.
[25] A. Kanda and D. Park, When are two effectively given domains identical?, *Proc. 4th GI Conf. on Theoretical Computer Science*, Aachen, Lecture Notes in Computer Science **67** (Springer, Berlin, 1979).
[26] S. Kleene, Countable functionals, in: A. Heyting, ed., *Constructivity in Mathematics* (North-Holland, Amsterdam, 1959).
[27] G. Kreisel, Interpretation of analysis by means of constructive functionals of finite types, in: A. Heyting, ed., *Constructivity in Mathematics* (North-Holland, Amsterdam, 1979).
[28] G. Longo, Hereditary partial effective functionals in any finite type, Preliminary Note, Forschungsinstitut f. Math., ETH Zürich, 1982.
[29] G. Longo, Set-theoretical models of lambda-calculus: Theories, expansions, isomorphisms, *Ann. Pure & Appl. Logic* (formerly: *Ann. Math. Logic*) **24** (1983) 153-188.
[30] G. Longo, Limits, higher type computability and type-free languages, *MFCS '84*, Prague, Lecture Notes in Computer Science **176** (Springer, Berlin, 1984) 96-114.
[31] G. Longo and E. Moggi, The hereditary partial effective functionals and recursion theory in higher types, *J. Symb. Logic* **49** (4) (1984) 127-140.
[32] G. Longo and E. Moggi, Gödel-numberings, principal morphisms, combinatory algebras, *MFCS '84*, Lecture Notes in Computer Science **176** (Springer, Berlin, 1984) 397-406.
[33] D. McQueen, G. Plotkin and R. Sethi, An ideal model for polymorphic types, Tech. Rept., Bell Lab., Murray Hill, 1983; *Inform. and Contr.*, to appear.
[34] A. Meyer, What is a model of lambda-calculus? *Inform. and Contr.* **52** (1) (1982) 87-122.
[35] R. Milner, A theory of type polymorphism in programming, *J. Comput. System Sci.* **17** (1978) 348-375.
[36] J. Myhill and C. Shepherdson, Effective operations on partial recursive functions, *Z. Math. Logik* **1** (1978) 310-317.
[37] D. Normann, *Recursion on the Countable Functionals*, Lecture Notes in Mathematics **811** (Springer, Berlin, 1980).
[38] G. Plotkin, LCF considered as a programming language, *Theoret. Comput. Sci.* **2** (1977) 189-210.
[39] G. Plotkin, $T\omega$ as a universal domain, *J. Comput. System Sci.* **17** (2) (1978) 209-236.
[40] H. Rogers, *Theory of Recursive Functions and Effective Computability* (McGraw-Hill, New York, 1969).
[41] D.S. Scott, Continuous lattices, in: E. Lawvère, ed., *Toposes, Algebraic Geometry and Logic*, Lecture Notes in Mathematics **274** (Springer, Berlin, 1972) 97-136.
[42] D.S. Scott, Data types as lattices, *SIAM J. Comput.* **5** (3) (1976) 522-587.
[43] D.S. Scott, Some ordered sets in Computer Science, in: I. Rival, ed., *Ordered sets* (Reidel, Dordrecht, The Netherlands, 1982).
[44] M. Smyth, Effectively given domains, *Theoret. Comput. Sci.* **5** (1977) 257-274.
[45] M. Smyth, Computability in categories, theory of computation, Rept., Univ. of Warwich, 1979.
[46] A. Troelstra, *Metamathematical Investigation of Intuitionistic Arithmetic and Analysis*, Lecture Notes in Mathematics **344** (Springer, Berlin, 1973).